A Structured Description Language for Multimedia Content Protection

C. Alberti[*], *A. Romeo*[*], *M. Mattavelli*[*], *C. Serrao*[**]
[*]Swiss Federal Institute of Technology - EPFL
Signal Processing Institute, ITS/LTS3, CH-1015 Lausanne, Switzerland
e-mail: claudio.alberti@epfl.ch
[**]ADETTI - Associação para o Desenvolvimento das Telecomunicações e Técnicas de Informática.
Edifício ISCTE, Av. Forcas Armadas, 1600-082 Lisboa, Portugal
e-mail: carlos.serrao@adetti.iscte.pt

ABSTRACT

In this paper a new approach to Intellectual Property Management and Protection (IPMP) of multimedia content streams is proposed. The innovative feature of the approach is that the protection procedure is associated, embedded and downloaded with the protected content that is "consumed" on standard terminals. Such an approach thus enables terminal interoperability, which is one of the main obstacles to the deployment of "secure" terminals. The approach takes its origin from the successful experiences in the standardization of toolsets for audio synthesis and processing in the framework of the Moving Picture Experts Group (MPEG). The approach is in course of development together with a prototype implementation in the framework of the IST project MOSES (MPEG Open Security for Embedded Systems).

1. INTRODUCTION

Nowadays, network infrastructures are starting to be used to support the commercialization of multimedia content as they are for many other commodities, like cars, domestic appliances and food. In fact, while the actual transactions for purchasing the goods can take place over a network infrastructure, like the Internet, the purchased items are physically delivered to the customers by traditional means (except for software). Examples of this approach are big on-line media stores like Amazon.com and other similar initiatives that have appeared in many countries, and which are gradually expanding their scope starting from book selling to include retailing of music on CD.

On one hand, it is clear that transactions of non-material goods, like books, videos, music, still images and any other type of mono-media or multi-media information share a number of peculiar qualities that are strongly in favor of their migration from traditional physical delivery technologies to an all-electronic delivery model: they can be digitized, stored, compressed and transferred on-line. On the other hand, in recent years consumers have been very reluctant to endorse this model used by market operators, especially content production companies. This attitude has shown not only as regards the Internet but also with respect to long established techniques for delivering digital audio-visual content, like the digital compact disk in the DVD version. This is mainly due to the fact that the same features that make the distribution and management of digital content so easy are also responsible for the difficulties of selling it on-line. Digital information as it is can be easily copied an unlimited number of times and transferred to an unlimited number of people.

Content protection of digital items has therefore become a major issue. On one hand content has to be protected so that access to it is enabled only to those who have acquired the right to do so, and on the other it has to be protected from uncontrolled dissemination.

One solution to this problem has been the development of proprietary "Intellectual Property" (IP) protection systems; but although they can be very effective for the specific platforms they are conceived for, they introduce significant interoperability limitations when trying to access the content using different devices.

In this case, interoperability among different manufacturers' products is of crucial importance in providing consumers with the easiest access to content.

This paper shows how the proposed approach is able to provide an effective way to securely handle multimedia content and at the same time to support two kinds of interoperability:

1. enable the same protected content to be consumed on different vendors' devices.
2. enable the same content to be protected by different vendors' IPMP tools.

The first step is the definition of a high-level meta-language for IPMP tool description. The core of such a language shall be a library of IPMP-primitives able to provide all the range of functionality required during content consumption. The standardization of such libraries will define the "normative" behaviour of any standard platform with no constraints on the implementation.

The interpreted nature of the suggested approach will allow on the one side an implementation-independent method of complexity evaluation and on the other interoperability among different technologies. This is achieved through different Virtual Machine (VM) binary codes running on the different platforms, but processing exactly the same IPMP information.

The interesting features of the approach described in this paper have been recognized by the MPEG committee by adding in the MPEG-4 IPMP draft one extension currently in development based on the structured description of IPMP tools described here.

2. STRUCTURED IPMP TOOLS

The idea of performing signal processing by means of an interpreted language has been already successfully exploited in the computer music community and brought to a multimedia international standard. In fact, the MPEG-4 Audio standard provides a toolset for synthetic Audio generation and Audio processing called Structured Audio (SA). SA permits the description of algorithms by means of the Structured Audio Orchestra Language (SAOL) programming language [3]. SAOL allows sound synthesis and process through instruments described as a network of primitives.

Following the essential idea behind SA, this paper outlines the proposal to define a meta-language based on a set of IPMP-primitives able to describe the widest range of content protection mechanisms including as a subset those deployed by the multimedia industry. Moreover, this language should be as flexible as possible in order to support the largest number of future techniques. For example, the core primitives shall be able to provide functionality needed when performing cryptographic algorithms. In case of asymmetric encryption/decryption, a RSA algorithm would be described in terms of a sequence of primitives (that henceforth will be called IPMP instrument) for receiving the cipher text as input and providing the plaintext as output after having retrieved the private key securely stored within the terminal or securely downloaded.

Every IPMP tool can be described by this language either as a single IPMP instrument or as a set of instruments composing an IPMP orchestra. The textual description shall then be formalized in terms of normative byte code defining the exact behaviour the terminal shall have in order to correctly process the information received. This implies that a given description of an IPMP instrument shall correspond to a unique byte code and that the entity interpreting the byte code has a corresponding unique normative behaviour. Its presence in any compliant terminal allows different developers to exchange IPMP tools without risk of incompatibility. Since the main task of this component will be to handle and coordinate the execution of several jobs at precise moments in time, hereafter it will be called a/the scheduler. Figure 1 shows how the textual description of an IPMP tool needs to be translated into a byte code in order to be downloaded into the terminal and executed by the normative scheduler.

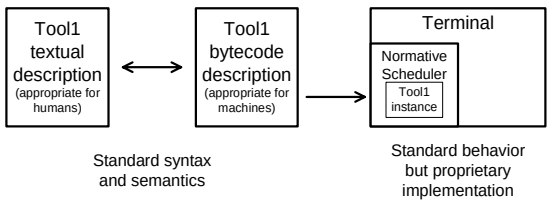

Figure 1. Tool description parsing, compiling and download into the terminal.

No implementation constraints are thus necessary except those concerning the secure implementation of the core execution engine of the VM. Any interpreter can be implemented with any available technology according to the business model for the content distribution it is conceived for, or according to the trade-off between security and implementation effort (costs).

The ability to describe IPMP tools in terms of sets of basic primitives employed a certain number of times during tool use will provide an implementation-independent method of complexity evaluation. In order to achieve meaningful measurements a complexity vector (i.e. a set of basic primitives or classes of primitives) can be flexibly defined according to the profiling needed. Every element of this vector will be a counter of the number of times a primitive (or primitives belonging to a class of primitives) is used. Complexity will be quantified through the values of the different vector dimensions. Platform independent complexity measurements will provide a consistent set of profiles applicable to any terminal and thus uniquely defining portability constraints.

3. DESCRIPTION OF TOOL RELATIONS

Many current usage scenarios imply the concurrent use of different IPMP tools implemented by different vendors. For instance the multimedia content can be first decrypted and then signed through some watermarking techniques The different tools involved are likely to be handled by different delivery infrastructure and will then be described by independent IPMP orchestras. The normative scheduler installed in any compliant terminal will then properly handle and coordinate the parallel (or serial) execution of the different protection mechanisms in order to obtain the desired result.

The authors propose that the behaviour of the scheduler during instantiation and execution of independent IPMP tools is formalized through a meta-language that can be referred to as IPMP Scene Description Language similar to the already existent and deployed MPEG-4 Binary Format for Scene (BIFS). The scene description can be coded independently from the streams related to primitive media objects so that it will not be necessary to decode the objects in order to access and if necessary modify parameters describing the scene.

In our view the different IPMP tools will play the role of BIFS nodes and their relationship will be described in terms of a hierarchical structure. Each node is linked to one or several other nodes in a non-static way. This means that node attributes can be changed while nodes can be added, replaced, or removed. This scenario further stresses the need for a normative scheduler able to coordinate the different IPMP tool instances running inside the virtual machine. The Open Platform Infrastructure for Multimedia Access (OPIMA) specification [1] seems to appropriately address this need by providing a starting point for the conception of this normative entity. Actually this architecture appears as the best candidate to interpret and

execute languages oriented to secure multimedia content handling. Some modifications to its original specification are likely to become necessary in order to best fit the set of primitives that will be defined and thus provide the maximum degree of effectiveness. For instance, one of the main open issues of the OPIMA architecture is the definition of a standard secure interface between IPMP tools and the terminal. An answer to this problem is provided by the MPEG IPMP Extensions (IPMPX) specification that standardizes a messaging interface allowing inter-IPMP tools and tools-terminal communication. An appropriate integration of the two solutions when implementing a prototype Structured IPMP decoder is currently going on within MOSES. This will hopefully represent the first answer to the need for an open and flexible framework for secure digital multimedia content handling.

As for tools description, scene description has to be formalized in a unique byte code to be downloaded into the terminal. Again, only the behaviour of the normative interpreter processing the scene description shall be standardized. No implementation constraints will be necessary since the different implementations will exclusively provide a standard input-output relation (a given description corresponds to a given byte code). Such description will allow the content vendor to exactly specify the way its property has to be used according to its favorite business models. A schematic of this scenario is shown in figure 2.

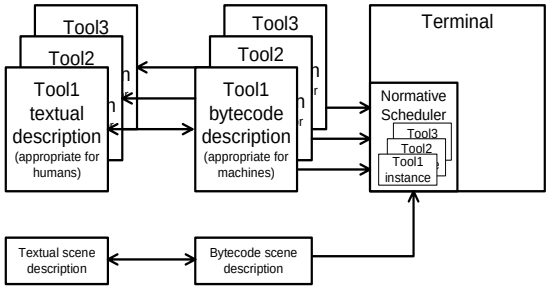

Figure 2 A process of content consumption may require several interoperating IPMP tools.

4. TOOLS RUN-TIME CONTROL

Once an IPMP instrument has been described in its functionality and placed in the correct place inside the scene, it has to be configured and used in the right way. This goal can be achieved by means of a score language providing information concerning: the exact time in which a particular IPMP instrument has to be instantiated; the duration of its performance (or the exact time in which it has to be terminated); the parameters needed for initialization.

For instance, if a process of content consumption requires the interaction with a remote IPMP tool, we can imagine that through a score event an IPMP instrument is instantiated and configured according to configuration parameters provided by the content vendor or trusted third parties. Then the instrument produces the appropriate message to be sent and stops. The remote tool, once the message is received, produces the answer and a score event aimed at activating the IPMP instrument instance recipient of the message. This provides an asynchronous mechanism of IPMP tool interaction.

By means of a score language like this access to some control variables inside the different IPMP instruments will be possible. Such control points are a sort of exposed (i.e. accessible for modification) variable that provide an additional description of the scene. This feature provides a powerful run-time tuning mechanism.

A score is a list of commands. A command performs a single action at a moment in time, such as changing the value of an exposed field or creating a new instance of an IPMP instrument (providing the set-up settings). The instantiation of a new IPMP tool only requires that the byte code related to the new tool description has already been downloaded into the terminal.

5. AN EXAMPLE OF STRUCTURED IPMP TOOL

A very simple example is provided in figure 3 so as to better illustrate the idea of IPMP tools structured description.

RSA

Algorithm: Key generation for RSA public-key encryption
SUMMARY: each entity creates an RSA public key and a corresponding private key. Each entity A should do the following:

1. Generate two large random (and distinct) primes p and q each roughly the same size.
2. Compute $n = pq$ and $\phi = (p-1)(q-1)$.
3. Select a random integer e, $1<e<\phi$, such that $\gcd(e,\phi)=1$.
4. Used the extended Euclidean algorithm to compute the unique integer d, $1<d<\phi$, such that $ed \equiv 1 \pmod{\phi}$.
5. A's public key is (n,e); A's private key is d.

Algorithm: RSA public-key encryption
SUMMARY: B encrypt a message m for A, which A decrypts.

1. *Encryption*. B should do the following:
   a. Obtain A's authentic public key (n,e).
   b. Represent the message as an integer m in the interval $[0, n-1]$.
   c. Compute $c=m^e \bmod n$
   d. Send the ciphertext c to A.
2. *Decryption*. To recover plaintext m from c, A should do the following:
   a. Use the private key d to recover $m=c^d \bmod n$.

```
RSA_Key_Generation(int KEY_LENGTH)
{
      bit(KEY_LENGHT/2)      p, q;
      bit(KEY_LENGHT)        d, e, n, φ ;

      while (p == q))
      {
            while(!is_prime(p))
                  p = select_random_pow2(1,KEY_LENGTH/2);
            while(!is_prime(q))
                  q = select_random_pow2(1,KEY_LENGTH/2);
      }
      φ=mul_big((p-1), (q-1))
      while(gcd(φ ,e) != 1) )
            e= select_random_pow2(1, φ);
      d=gcd_ext(φ ,e,a,b);
      n=p*q
      return (n,e,d);
}

RSA_Encryption(bit[] m, bit[] e, bit[] n, int KEY_LENGTH)
{
      bit(KEY_LENGTH) c;
      c=exp_mod(m,e,n);
      return(c);
}

RSA_Decryption(bit[] c, bit[] d, bit[] n, int KEY_LENGTH)
{
      bit(KEY_LENGTH) m;
      m=exp_mod(c,d,n);
      return (m)
}
```

Figure 3 Example of RSA algorithm described by means of the structured IPMP approach.

The described tool performs the RSA (named after its inventors R. Rivest, A. Shamir, L. Adelman) public-key encryption algorithm. In bold are functions that could belong to the primitive instructions set to be defined. The description is taken from [4] where the necessary keys are generated and then used to encrypt and decrypt the sensible content.

The description of the RSA algorithm provided here permit the definition of a first set of primitives that are likely to be inserted into the core library of functions: operations modulo a generic integer *n* (*exp_mod*), operations on great numbers (*mul_big*), operations on random and prime numbers (*is_prime, select_random_pow2*). The study and analysis of the widest range of currently used algorithms will assure the core primitives cover the largest variety of usage scenarios so as to be able to implement virtually any possible protection tool.

6. CONCLUSIONS

As the results of SA implementations have shown [2] [5] [6], the interpreted approach provides a consistent number of advantages that we summarize here.

The structured tool description by means of a network of primitives can allow the implementation of the widest range of tools if the definition of the core set is appropriate and the language is flexible; it provides a simple way to redesign IPMP tools once they could become untrustworthy because of implementation flaws or algorithm intrinsic weaknesses; in case of remote tools it does not need the protected content to exit the terminal since any structured IPMP tool is downloadable and the only data transfer involved is that concerning the byte code download; it allows a precise complexity evaluation in terms of number of basic instruction to be performed per time-frame and thus implementation independent. These complexity measurements lead to a precise definition of terminals in terms of processing capabilities.

The scene description in terms of dedicated language provides a simple way to describe the interaction between different IPMP tools even after the content consumption has started; since it is coded in textual format and downloaded as normative byte code, it requires a low bit-rate data exchange with the terminal, keeping any sensible information inside the trusted compartment; it implies the definition of a normative behaviour for the entity supervising the overall execution with no constraint about the implementation.

The tool configuration by means of a dedicated score language allows the on-the-fly instantiation and (re)design of new IPMP tools through events generation and is appropriate for a precise and direct control of IPMP tools performances thanks to the exposed fields mechanism.

It may be argued that there is no need to have a new language to define structured descriptions of IPMP tools and a language such as Java could be employed for such goal. In reality there are several advantages in defining an appropriate language in conjunction with an appropriate scheduler (OPIMA [1]):

The simplicity of the language allows us to accurately takie into account the widest range of possible security issues (something that is not possible – or unfeasible – with a general purpose language like Java);

The specific libraries and functions can be implemented through optimized native code routines or exploiting already existent hardware accelerators;

The security of the VM can be achieved through the latest techniques of secure software production [7], the deployment of technologies for binary code protection such as the “Tamper Resistant Coding” [8] or using secure hardware components (crypto accelerators, secure storage devices etc.).

7. REFERENCES

[1] Open Platform Initiative for Multimedia Access Specification Version 1.1. Turin, 2000.

[2] G. Zoia, *A Virtual Model for Simulation and Design of Architectures in MPEG-4 Audio and Multimedia Context*, Ph. D. thesis No. 2362, Swiss Federal Institute of Technology, Lausanne, April 2001, Chapter 5.

[3] E. Scheirer, *SAOL: the MPEG-4 Structured Audio Orchestra Language*, Proceedings of the International Computer Music Conference. Ann Arbor, MI, October 1998.

[4] A.J. Menezes, P.C. van Oorschot, S. A. van Stone, *Handbook of Applied Cryptography*, CRC Press, Chapter 8.2.

[5] G. Zoia, C. Alberti, *An Efficient Block-Based Interpreter for MPEG-4 Structured Audio*, Proceedings of the IEEE Int Symp. on Circuits and Systems - ISCAS 2000, Geneva, CH, May 2000.

[6] G. Zoia, C. Alberti, *A virtual DSP architecture for MPEG-4 Structured Audio*, Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00), Verona, Italy, December 7-9, 2000.

[7] J. Viega, G. McGraw, *Building Secure Software*. Addison-Wesley, 2001.

[8] K. Saito, *What is "Tamper Resistant Coding Technology"?*, http://www.accessticket.com/eng-trc1.html, 2000.